APPENDIX

Photolithography Overlay Map Generation with Implicit Knowledge Distillation Diffusion Transformer

A. Background

A.1. Overlay Misregistration Errors Introduction

Lithography overlay misregistration errors are quantified using the error vector magnitude (EVM) [1], assessed after the photoresist coating and exposure processes to measure the misalignment between the current and preceding circuit maps, as depicted in Fig. 1. EVM is defined as the vector difference between the lithography stacking vector V_{GT} , obtained from actual overlay measurements across all wafers per lot, and the predicted stacking vector V_P from AI models like GAGAN [1] and our proposed IKDDiT, illustrated in Fig. 2. The formula for EVM is given by:

$$EVM = \sqrt{X_{err}^2 + Y_{err}^2} \qquad (1)$$

where X_{err} and Y_{err} denote the error components along the X-axis and Y-axis, respectively, reflecting deviations in the lithography overlay process. As shown in Fig. 2, these components are orthogonal projections of the EVM vector, representing the misalignment between the actual and predicted lithography stacking vectors V_{GT} and V_P. In this work, EVM is employed as a metric instead of traditional pixel-based measures such as PSNR.

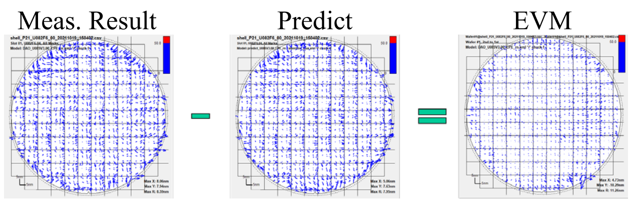


Figure 1: Overlay misregistration errors on wafer map [1].

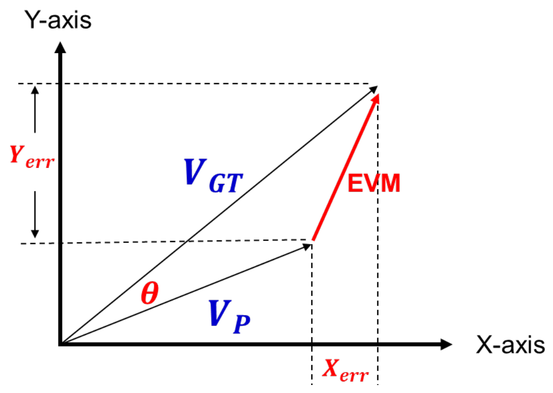


Figure 2: Illustration of error vector magnitude calculation [1].

A.2. Business Impact

The impact on production capacity caused by overlay pilot runs can be demonstrated in Fig. 3 [1]. As illustrated in Fig. 3a, it is essential to split two wafers from a single lot to conduct an overlay pilot run, enabling the determination of overlay misregistration errors. These two wafers undergo rework (removal of photoresist during the etching process), are returned to the original process step, and subsequently merged with the parent lot. Finally, the second production process is performed on the entire batch of 25 wafers, employing feed-forward error compensation based on the average overlay measurements of the two pilot wafers. Reworking each batch significantly impacts lithography equipment efficiency, resulting in a loss of scanner productivity.

In Fig. 3b, the APC system can perform feed-forward error compensation through AI model predictions of the overlay map. This eliminates the need to split lots for pilot runs, thus avoiding scanner loss and enhancing productivity. Lithography is widely recognized as the bottleneck in semiconductor manufacturing processes. Therefore, improving the productivity of lithography equipment directly enhances the overall productivity of the foundry, leading to substantial revenue increases for semiconductor enterprise.

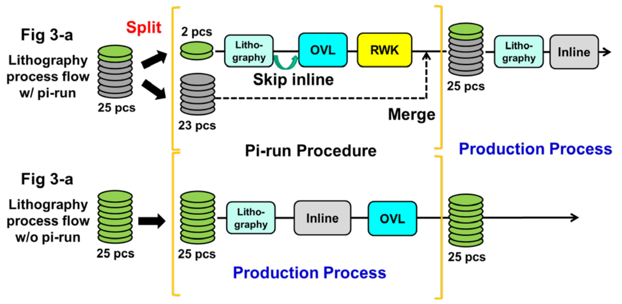


Figure 3. Lithography process flow w/ & w/o pilot run [1].

A.3. Limitations

While IKDDiT is a specialized generative AI model developed for overlay map generation in the semiconductor manufacturing industry, its capabilities extend beyond this intended use, raising potential concerns about misuse. For instance, IKDDiT can be leveraged to produce fake overlay maps that may deceive quality control systems or disrupt manufacturing processes if used with malicious intent. Recognizing these risks, access to

the model is restricted, and its use is carefully controlled to prevent exploitation.

Moreover, the underlying architecture of IKDDiT, with its advanced generative capabilities, could be adapted to generate synthetic images or even manipulated visuals, such as those used in fake news or misleading media. The power of this technology highlights the need for strict governance, ethical considerations, and cautious management. Ensuring that IKDDiT is only used in legitimate, monitored settings is crucial to prevent potential societal harm that could arise from its ability to generate highly realistic yet fabricated content.

B. Implementation Details

B.1. Diffusion Configurations

Unlike the DiT [2], which relies on the ADM framework introduced by Dhariwal and Nichol [3], we employ the EDM framework [4] to streamline training and enhance inference efficiency. In particular, we utilize EDM preconditioning through a σ-dependent skip connection, which involves carefully calibrated hyperparameters [4]. This approach allows us to bypass ADM's noise covariance parameterization as implemented in DiT. During inference, we follow a default time sequence:

$$t_i < N = \left(\frac{1}{t_{max}} + \frac{i}{N-1}\left(t_{min}^{\frac{1}{\rho}} - t_{max}^{\frac{1}{\rho}}\right)\right)^{\rho} \qquad (2)$$

where $N = 40$, $\rho = 7$, $t_{max} = 80$, and $t_{min} = 0.002$. Additionally, we adopt Heun's method, a numerical procedure for solving ordinary differential equations (ODEs) with a given initial value, as the ODEs solver for sampling, which, according to our preliminary tests, achieves the same FID as the 200 sampling steps used in DiT but requires significantly fewer function evaluations (60 evaluations instead of 200), thus supporting the findings by [3].

B.2. KIDDiT Architecture

Our IKDDiT model adheres to the standard DiT architecture, comprising a series of Gated Cross-Attention DiT blocks. Each block integrates a multi-head cross-attention module, tanh gating, a multi-head self-attention module, and a pointwise feedforward layer. The teacher decoder concludes with layer normalization. Since the encoder and decoder in IKDDiT have different weights, we employ a linear projection layer after the decoder to align them. Additionally, IKDDiT incorporates sine-cosine positional embeddings [5] for the teacher encoder, student encoder, and student decoder inputs. Notably, our design does not use relative position embeddings or layer scaling.

The teacher-student architecture effectively doubles the parameter count compared to a standard DiT model. However, during inference, the teacher network is discarded, meaning there is no additional parameter burden. Consequently, the learned KiDDiT-XL model has a size of 762 million parameters, which is on par with MaskDiT-XL [6] (730 million). Although KiDDiT-XL has a slightly larger model size than SD-DiT-XL [7] (741 million), however, KiDDiT-XL is not designed to be deployed on edge devices, so this model size is acceptable.

Additionally, during training, SD-DiT employs an exponentially moving average (EMA) to update the teacher encoder without using backpropagation via SGD, resulting in minimal computational overhead. In contrast, IKiDDiT trains the teacher encoder's parameters independently, incurring higher computational costs. Nevertheless, similar to SD-DiT, IKiDDiT removes the teacher encoder entirely during inference, thereby avoiding any additional load.

B.3. Compare Model Settings

DiT [2] include variations at both 256 × 256 and 512 × 512 resolutions, with our experiments focusing on the 512 × 512 resolution models for performance comparisons. To encode input time steps, we employ a 256-dimensional frequency embedding, followed by a two-layer MLP. This MLP is designed to match the transformer's hidden layer size and uses SiLU activations. The AdaLN layer integrates the time step and class embeddings, applying SiLU nonlinearity and a linear transformation. For the core transformer operations, DiT use GELU activations, approximated with tanh in IKDDiT. Additionally, we implement classifier-free guidance on selected channels.

MDT. [8] features a architecture comprising 28 transformer layers, each with a hidden dimension of 1152 and 16 attention heads. This design yields a model with 675.8 million parameters and a computational complexity of 118.69 GFLOPs. Consistent with the established design principles of MDT, we adopt a smaller patch size of p = 2, a choice demonstrated to enhance synthesis performance and improve the generation of fine details.

MaskDiT [6] employs an asymmetric encoder-decoder architecture. The encoder is based on DiT-XL, the largest model configuration from DiT, using a patch size of 2 for all input patches. It retains the original DiT architecture but omits the final linear projection layer and processes only unmasked patches. Consistent with DiT, the encoder maps patches through a linear projection, adding standard ViT frequency-based positional embeddings to all input tokens. Masked tokens are subsequently removed before processing the remaining encoder layers.

The decoder follows the architecture of MAE [9] with modifications: adaptive layer normalization blocks are added to condition on time and class embeddings. This adapted decoder, derived from a lightweight MAE design, features only two DiT blocks and operates on the complete set of tokens.

SD-DiT [7] integrates the DiT [2] block as the fundamental transformer unit within its backbone network, incorporating adaptive layer normalization to merge conditional time and class embeddings. Like MaskDiT, SD-DiT employs an asymmetric encoder-decoder structure for the generative diffusion process. The student DiT encoder is configured as DiT-XL with a patch size of 2, while the compact DiT decoder, comprising 8 DiT blocks, mirrors the design principles of MAE.

For the discriminative objective, SD-DiT adopts configurations inspired by iBOT [10] and DINO [11]. The teacher DiT encoder functions as an exponential moving average (EMA) of the student encoder, with the momentum coefficient progressively increasing from 0.996 to 0.999 by the end of training.

B.4. Training Details

VAE Pre-training. The VAE pre-training settings are summarized in Table 1. We employ the AdamW [12] optimizer with a base learning rate of 0.0015, combined with a weight decay of 0.05. The optimizer's momentum parameters are set to β_1=0.9 and β_2=0.95. Training is conducted with a batch size of 4096, using a cosine decay learning rate schedule [13] and 5 warmup epochs over 50 total training epochs. Data augmentation methods include Random Resized Crop, Color Jittering, and Gradient Clipping.

Image-Text Encoder Pre-training. The pre-training configuration for the Image-Text Encoder by unified contrastive learning is also presented in Table 1. We also use AdamW as the optimizer, this time with a base learning rate of 0.003 and the same weight decay of 0.05. The momentum parameters remain β_1=0.9 and β_2=0.95, with a reduced batch size of 1024. The learning rate schedule again employs cosine decay with 5 warmup epochs over 50 training epochs. No additional augmentation is applied.

IKDDiT Training. For end-to-end training of the IKDDiT model, we use the AdamW optimizer with a base learning rate of 0.003, weight decay of 0.05, and momentum parameters adjusted to β_1=0.9 and β_2=0.95. The batch size is significantly reduced to 64 to accommodate end-to-end fine-tuning. The learning rate schedule remains cosine decay with 10 warmup epochs, extending to 100 training epochs. The only augmentation used in this phase is random resized cropping.

B.5. Computing Resource Configuration

The comparison of training efficiency for DiT, MDT, MaskDiT, SD-DiT, and IKDDiT was conducted using PyTorch version 1.12.0 on a setup comprising 8× NVIDIA A100 80 GB GPUs. All models were evaluated at the XL scale to maintain consistency. The experiments utilized a training batch size of 64, and each model was trained for 578.1k iterations, ensuring a comprehensive and uniform assessment of performance across different architectures.

Table 1: VAE pre-training setting.

Config	Value
Optimizer	AdamW
Base Learning Rate	0.0015
Weight Decay	0.05
Optimizer Momentum	β_1=0.9, β_2=0.95
Batch Size	4096
Learning Rate Schedule	Cosine Decay
Warmup Epochs	5
Training Epochs	50
Augmentation	RandomResizedCrop Color Jittering Gradient Clipping

Table 2: Image-Text encoder pre-training setting.

Config	Value
Optimizer	AdamW
Base Learning Rate	0.003
Weight Decay	0.05
Optimizer Momentum	β_1=0.9, β_2=0.95
Batch Size	1024
Learning Rate Schedule	Cosine Decay
Warmup Epochs	5
Training Epochs	50
Augmentation	N/A

Table 3: IKDDiT end-to-end training setting.

Config	Value
Optimizer	AdamW
Base Learning Rate	0.003
Weight Decay	0.05
Optimizer Momentum	β_1=0.9, β_2=0.999
Batch Size	64
Learning Rate Schedule	Cosine Decay
Warmup Epochs	10
Training Epochs	100
Augmentation	RandomResizedCrop

C. Preliminaries

The primary generative framework of this paper is the Diffusion Model. We will first introduce the preliminaries of Diffusion Models, followed by an explanation of Latent Diffusion Models, which perform the diffusion and denoising processes in the latent space. Lastly, we will discuss the concept of classifier-free guidance.

C.1. Diffusion Model

To lay the groundwork for our architectural design, we begin by summarizing essential concepts relevant to diffusion probabilistic models (DDPMs) [14]. These

models generate an image by iteratively reversing a gradual process of introducing noise. The image x_0, which follows a distribution $q(x_0)$, is corrupted by the forward process q through the gradual addition of Gaussian noises over T steps.

$$q(x_t|x_{t-1}) := \mathcal{N}\left(x_t; \sqrt{1-\beta_t}x_{t-1}, \beta_t I\right) \tag{3}$$

where β_t determines the variance of noises added at timestep t. The forward process results in a sequence of latent variables x_1, ..., x_T that become increasingly noisy. Eventually, with enough iterations, this sequence reaches a state where it consists purely of noise, specifically $x_T \sim N(0, I)$. Importantly, it is possible to marginalize the intermediate steps and directly derive x_t from x_0.

$$q(x_t|x_0) := \mathcal{N}\left(x_t; \sqrt{\bar{\alpha}_t}x_0, (1-\bar{\alpha}_t)I\right) \tag{4}$$

where $\alpha_t := 1 - \beta_t$ and $\bar{\alpha}_t := \prod_{s=1}^{t} \alpha_s$. $\bar{\alpha}_t$ are predetermined hyperparameters. Using the reparameterization trick, we derive samples as:

$$x_t(x_0, \epsilon) = \sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon \tag{5}$$

where ϵ is the standard Gaussian noise. The core objective of diffusion models is to approximate the reverse denoising process, represented as:

$$p_\theta(x_{t-1}|x_t) := \mathcal{N}(x_{t-1}; \mu_\theta(x_t, t), \sigma_t^2 \mathrm{I}) \tag{6}$$

where neural networks are leveraged to estimate the parameters of p_θ . The reverse model is optimized via a variational lower bound on the data log-likelihood x_0, formulated as:

$$\mathcal{L}(\theta) = -\log p_\theta(x_{t-1}|x_t) \\ + \sum_t D_{KL}(q^*(x_{t-1}|x_t, x_0)||p_\theta(x_{t-1}|x_t)) \tag{7}$$

where excluding an irrelevant constant term for training. Since q^* and p_θ are Gaussian, the Kullback-Leibler divergence D_{KL} can be computed using the means and covariances. Reparameterizing μ_θ with a noise prediction model ϵ_θ, we simplify training to minimize the error between predicted noise $\epsilon_\theta(x_t)$ and the ϵ_t ground truth Gaussian noise:

$$\mathcal{L}_{simple} = \mathbb{E}_{x,\epsilon,t}[||\epsilon_t - \epsilon_\theta(x_t(x_0, \epsilon), t)||_2^2] \tag{8}$$

Nevertheless, the diffusion model also needs optimization of the reverse process variance Σ_θ, which makes the full objective $\mathcal{L}_{KL}$ necessary. Adopting the method of Nichol and Dhariwal [15], we pre-train ϵ_θ using $\mathcal{L}_{simple}$ and subsequently train Σ_θ with the full objective. The model inference starts from $x_t \sim \mathcal{N}(0, \mathrm{I})$ and generates samples via $x_{t-1} \sim p_\theta(x_{t-1}|x_t)$, utilizing the reparameterization trick.

C.2. Latent Diffusion Models

Training diffusion models directly in high-resolution pixel space often results in computational challenges due to prohibitive costs. Latent Diffusion Models (LDMs) [16] offer an efficient solution with a two-step process: (1) using an autoencoder to condense images into a lower-dimensional latent space with a learned encoder E; (2) training a diffusion model on these compact latent representations $z = E(x)$, rather than the full-resolution image x (where E remains fixed). Images can then be synthesized by drawing samples z from the diffusion model and reconstructing them using the decoder $x = D(z)$.

As depicted in Figure 2, LDMs demonstrate strong performance while utilizing only a fraction of the computational resources needed by pixel-space diffusion models like ADM. This efficiency in terms of GFLOPs makes LDMs an attractive foundation for exploring architectural advancements. In our work, we adapt IKDDiT and other masked DiTs for latent space operation, though they can also be extended to pixel space without any modifications. This results in a hybrid approach for image generation, employing both convolutional VAEs and transformer-based DDPMs.

C.3. Classifier-free Guidance

Conditional diffusion models typically incorporate additional input information, such as class labels c. Under these circumstances, the reverse diffusion process is defined as $p_\theta(x_{t-1}|x_t, c)$, where both ϵ_θ and Σ_θ are conditioned on c. In this context, classifier-free guidance can be employed to direct the sampling procedure in finding x that maximizes the log-likelihood $\log p(c|x)$ [17]. Using Bayes' Rule, we derive $\nabla_x \log p(c|x) \propto \nabla_x \log p(x|c) - \nabla_x \log p(x)$. By interpreting the output of diffusion models as a score function, the sampling process in DDPMs can be steered to sample x with high $p(x|c)$ using:

$$\epsilon_\theta(x_t, c) = \epsilon_\theta(x_t, \emptyset) + \omega \cdot \nabla_x \log p(c|x) \tag{9}$$

where ω represents the guidance scale, where $\omega = 1$ denotes standard sampling without any guidance. To train the model without conditioning, we randomly drop c during training and replace it with a learned "null" embedding $\emptyset$. Classifier-free guidance is well-documented to significantly enhance sample quality compared to traditional sampling methods [14], and this trend is evident in our IKDDiT models as well.

MUSE [18] utilizes a dynamic approach by progressively increasing the guidance scale linearly throughout the sampling process. This technique enhances sample diversity in the initial stages while ensuring higher fidelity as sampling progresses. Building on this concept, we introduce a power-cosine schedule for adjusting the guidance scale during sampling:

$$\omega_t = \frac{1-\cos \pi \left(\frac{t}{t_{max}}\right)^s}{2} \omega \tag{10}$$

In this context, t denotes the time step within the sampling process, t_{max} represents the total number of sampling steps, ω specifies the peak guidance scale, and s adjusts the rate at which the guidance scale increases. As illustrated in Fig. 4, the power-cosine schedule maintains a lower guidance scale in the initial steps, followed by a rapid escalation as the process advances. A higher s value slows the increase at the beginning and accelerates it near the end. This enhanced classifier-free guidance approach, utilizing the power-cosine schedule, promotes sample diversity in the early stages while ensuring superior quality in the later stages. For our experiments, s is set to 4 and ω to 4, optimizing image fidelity in the final steps.

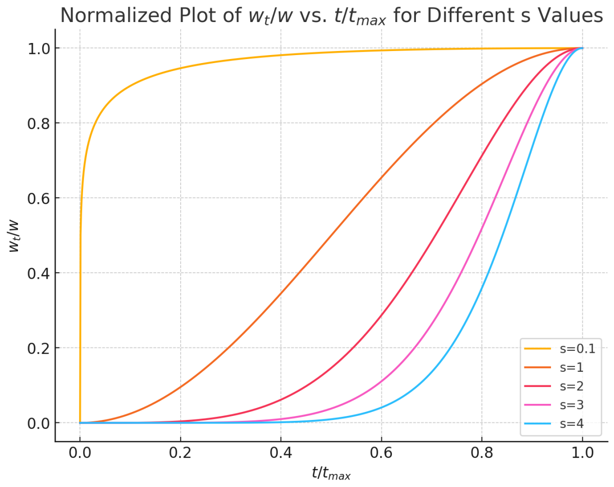


Figure 4. The power-cosine scaling schedule applied to the guidance scale in classifier-free guidance for varying values of s. When s is larger, the increase of ω progresses more gradually during the initial steps and accelerates more rapidly in the later steps.

D. Attention Mechanisms

This section delves into an in-depth exploration of the attention mechanisms in other masked DiT, encompassing the Multi-Head Self-Attention and Adaptive Normalization Layer Zero Block (adaLN-Zero block) introduced by DiT, and the Positional-Aware Self-Attention proposed by MDT.

D.1. Multi-Head Self-Attention

Multi-Head Self-Attention mechanism integrates the embeddings of t and c into a two-element sequence, kept distinct from the sequence of image tokens. The transformer block is adjusted to incorporate an additional multi-head cross-attention layer after the self-attention block, drawing inspiration from the original design by Vaswani et al. [5] and akin to the implementation in LDMs [16] for class label conditioning. This cross-attention component significantly increases the computational cost, contributing approximately a 15% increase in Gflops.

D.2. Adaptive Normalization Layer Zero Block

Peebles and Xie [2] explore replacing standard layer normalization layers in transformer blocks with adaptive layer normalization (adaLN). Instead of directly learning the scale and shift parameters γ and β for each dimension, adaLN computes these parameters by regressing from the sum of embedding vectors t and c. Building on this, Peebles and Xie [2] further modify the adaLN DiT block, naming it the adaLN block, and extend the method to also regress dimension-specific scaling parameters α . These α parameters are applied before residual connections in the DiT block, with the MLP outputting α initialized to produce a zero vector, ensuring the DiT block functions as an identity mapping initially.

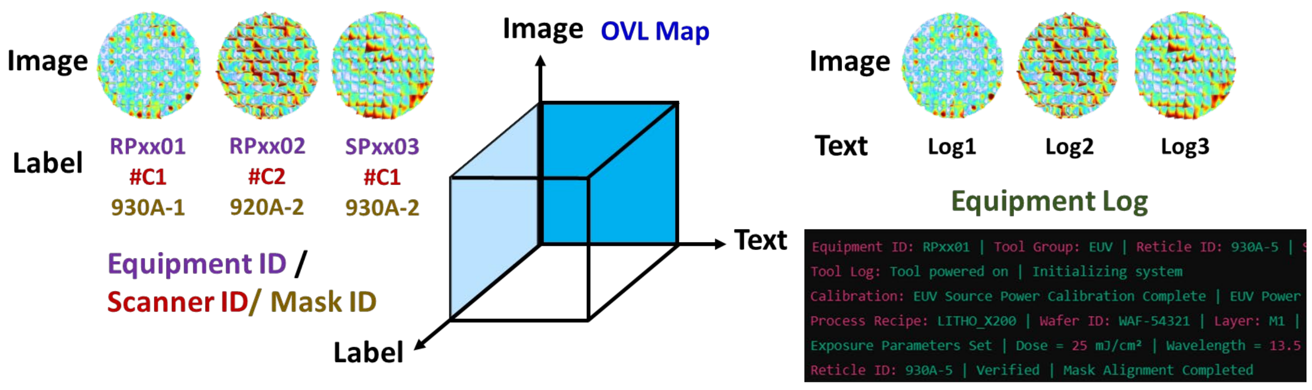
D.3. Positional-Aware Self-Attention

To enhance positional information in the model, Gao et al. propose a Positional-Aware Self-Attention mechanism for both the encoder and decoder, facilitating the learning of masked latent tokens. This mechanism leverages positional relationships among all tokens, allowing the model to predict masked latent tokens using the unmasked ones effectively. The encoder adds conventional learnable global position embeddings to the noisy latent input, while the decoder applies position embeddings differently during training and inference. During training, a side-interpolator integrates the global position embeddings, whereas in the inference phase, the side-interpolator is removed, and the decoder explicitly incorporates position embeddings to retain the enriched positional context.

Secondly, each block of the encoder and decoder incorporates a local relative positional bias for every head when computing self-attention scores. Specifically,

$$Attention(Q,K,V) = softmax\left(\frac{QK^\intercal}{\sqrt{d_k}} + B_r\right) \tag{11}$$

where Q, K, V represent the query, key, and value in the self-attention module, respectively; d_k denotes the dimension of the key, and B_r is the relative positional bias. The relative positional bias B_r is determined by the difference between the i-th position and other positions and is updated during training. This bias captures local positional relationships, enhancing the modeling of masked latent tokens. The relative positional bias B_r is determined by the difference between the i-th position and other positions and is updated during training. This bias captures local positional relationships, enhancing the modeling of masked latent tokens.



```
Equipment ID: RPxx01 | Tool Group: EUV | Reticle ID: 930A-5 | S
Tool Log: Tool powered on | Initializing system
Calibration: EUV Source Power Calibration Complete | EUV Power
Process Recipe: LITHO_X200 | Wafer ID: WAF-54321 | Layer: M1 |
Exposure Parameters Set | Dose = 25 mJ/cm² | Wavelength = 13.5
Reticle ID: 930A-5 | Verified | Mask Alignment Completed
```

Figure 5: Unified contrastive learning framework in the image-text-label space, seamlessly integrating supervised learning with image-label data and language-image contrastive learning using image-text data.

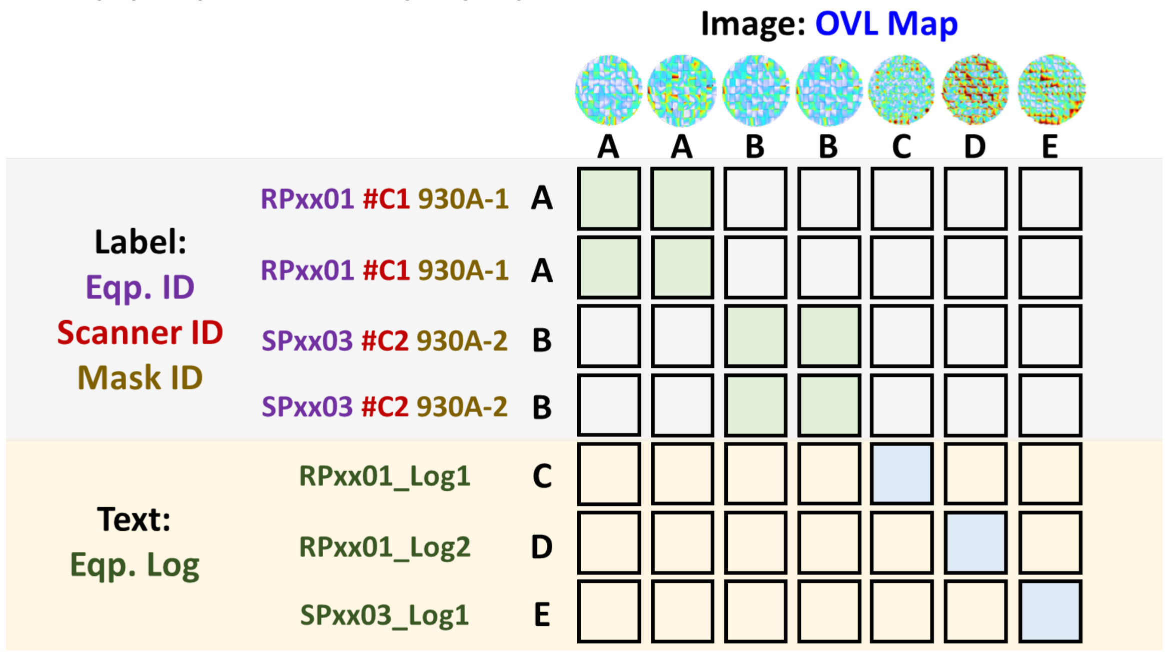


Figure 6: An illustration of covering image-label and image-text data in the image-text-label space.

E. Unified Contrastive Learning

In IKDDiT, we implement a unified contrastive learning paradigm [19] designed to train an image-text discriminative model that seamlessly integrates visual, textual, and categorical data within a shared embedding space. This comprehensive approach extends beyond conventional pairwise contrastive learning by harmonizing semantic information from various modalities, including images, text descriptions (equipment logs), and categorical labels (e.g., equipment, scanner exposure chuck, and reticle IDs). Such integration allows for more robust cross-modal representation learning, capturing the complex relationships inherent in multi-source industrial data.

Figure 5 provides a visual representation of this unified framework, showcasing how supervised learning on image-label pairs is combined with language-image contrastive learning using image-text pairs. This hybrid learning setup ensures that the model can learn rich and meaningful embeddings by leveraging both types of data. Figure 6 elaborates on the intricate organization of image-label and image-text data within the image-text-label space, illustrating how these modalities are effectively aligned through image-text-label triplets. The diagram emphasizes the distinction between positive and negative pairs, with positive pairs highlighted using green and blue tiles and negative pairs left uncolored.

Image-label data pair images with textual concepts based on annotated labels, shown as green tiles for semantic alignment. Image-text data are uniquely indexed, matching only at diagonal entries, marked by blue tiles. This setup highlights the model's precision in learning cross-modal relationships, boosting downstream task performance.

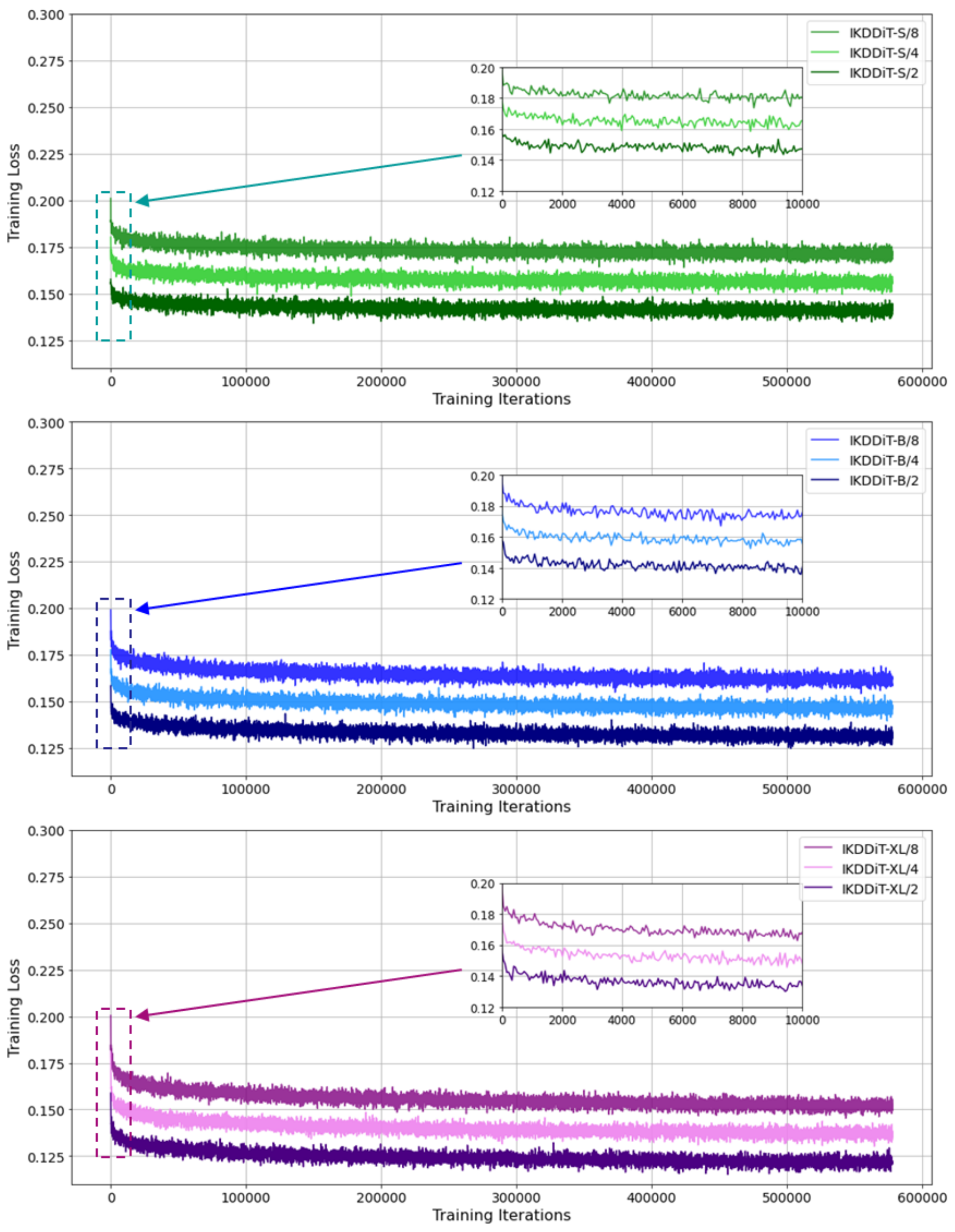


Figure 7: Training loss curves for all IKDDiT models are presented, capturing the loss progression recorded every 1000 iterations throughout the training period. We compare the performance across different model sizes: IKDDiT-S, IKDDiT-B, and IKDDiT-LX, which correspond to small, base, and extra-large configurations, respectively.

F. Model Scaling on Training Loss

Figure 7 illustrates the impact of scaling on training loss, focusing on how increasing the computational complexity of the IKDDiT model influences convergence behavior. We plot the loss over training for all IKDDiT models (the sum of the noise prediction mean-squared error and D_{KL}). By expanding the transformer architecture, whether through adding more layers, enlarging hidden state dimensions, or increasing the number of input tokens, the model achieves a steeper decline in training loss, resulting in faster convergence and a lower final loss value compared to smaller models. This trend is consistent with observations in language modeling, where larger transformer architectures not only demonstrate improved loss curves but also deliver superior performance across various downstream evaluation benchmarks [20]. These findings highlight the advantages of model scaling in enhancing learning efficiency and generalization, offering valuable insights for designing high-performance models suitable for complex tasks.

G. Additional Experiments

In this section, we conduct additional experiments to explore the optimal model configurations and parameters, focusing on classifier-free guidance scores, positional-aware attention mechanisms, interactions between full and partially unmasked latent tokens, VAE decoders, and varying masking ratios. Experiments were performed using the XL model configuration with default training settings, employing FID as the primary evaluation metric. To expedite assessments, we reduced training iterations to 150k.

G.1. Classifier-free Guidance Score

In this study, we implement a power-cosine schedule [18], which linearly increases the guidance scale throughout the sampling process, as described in Eq. 10 in Section C-3. This approach enhances diversity in the initial sampling stages while gradually ensuring higher fidelity as sampling progresses. We experimented with five different guidance scale scores ω (3, 3.5, 4, 4.5, and 5) and five guidance scale rates s (0.1, 1, 2, 3, and 4) to comprehensively analyze their impact on the model's performance.

As shown in Table 4, the experiment reveals that the model achieves its best performance with a guidance scale $\omega = 4$ and a guidance rate $s = 4$, yielding the lowest FID score of 24.66. This indicates that a moderate guidance scale combined with a progressively increasing guidance rate effectively balances sample diversity and generation quality, enabling the model to reach optimal performance during the sampling process.

When the guidance scale w is too small (less than 4), the model's output is less influenced by the conditioning input, resulting in noisier and less accurate samples. On the other hand, if the guidance scale w is too large (greater than 4), the generated samples become overly uniform, with limited diversity. In such cases, the images generated exhibit very limited variation, stripping the model of the inherent flexibility expected of a generative model and potentially producing samples that almost exactly replicate the training data, thus diminishing the creative aspect of the generation process. This extreme setting also introduces the risk of overfitting, where the model becomes overly tailored to specific conditioning inputs from the training data, lacking natural random variability and ultimately impairing the model's generalization capability.

Therefore, we adopt a guidance scale $\omega = 4$ and a guidance rate $s = 4$ as the Classifier-Free Guidance hyperparameters for IKDDiT's Diffusion Model, ensuring a well-balanced and high-quality generative performance.

G.2. Positional-aware Attention

In this section, we evaluate the performance of two attention mechanisms: the conventional multi-head self-attention and the Positional-Aware Self-Attention. As

Table 4: Ablation study on the classifier-free guidance score.

ω	s				
	0.1	1	2	3	4
3	39.01	34.79	32.61	30.41	29.74
3.5	35.67	31.89	29.61	27.73	27.10
4	31.80	28.18	26.61	25.24	**24.66**
4.5	32.64	29.49	27.41	25.21	24.71
5	34.44	30.45	28.77	26.59	26.03

Table 5: Ablation study on positional-aware attention.

Self-Attention	FID-15k
with Br	**24.66**
w/o Br	33.03

Table 6: Ablation study on the full and unmasked latent tokens.

Latent Embeddings	FID-15k
Unmasked Tokens	37.60
Full Tokens	25.66
Unmasked Tokens+Full Tokens	**24.66**

Table 7: Ablation study on the VAE decoders.

Decoder	FID-15k
MSE	24.99
EMA	**24.66**

Table 8: Ablation study on the masking ratio.

Mask Ratio	0	0.1	0.2	0.3	0.4
FID-15k	26.28	25.53	26.38	26.12	25.43
Mask Ratio	0.5	0.6	0.7	0.8	0.9
FID-15k	**24.66**	25.33	25.88	27.09	30.51

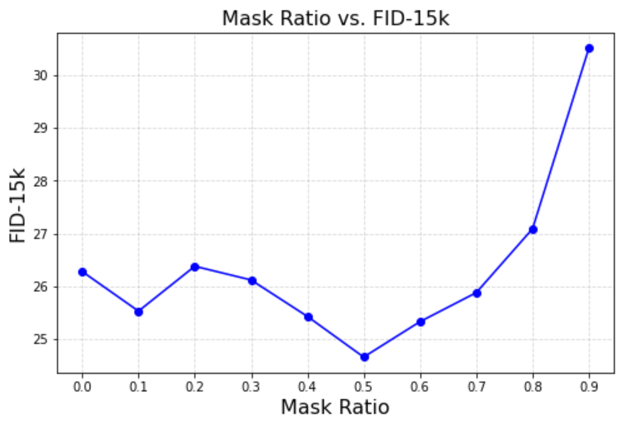


Figure 8: Masking ratio and FIG-15k

described in Eq. 11, Positional-Aware Self-Attention extends traditional self-attention by incorporating an additional term, B_r, which captures positional relationships among all tokens. From the experimental results shown in Table 5, it is evident that integrating B_r into the multi-head self-attention mechanism of IKDDiT significantly enhances generation quality. Consequently, we adopt Positional-Aware Self-Attention in the teacher encoder, student encoder, and student decoder.

G.3. Full and Unmasked Latent Tokens

In IKDDiT, both full and unmasked latent embeddings are input into the diffusion model during training. For comparison, we also trained models using only full latent embeddings and only unmasked latent embeddings, as shown in Table 6. The results demonstrate that training with both full and remaining unmasked latent embeddings yields a clear improvement over the two alternative approaches. Using only the unmasked latent embeddings results in slower convergence, which we attribute to the inconsistency between training and inference processes, as inference in IKDDiT is based on diffusion rather than a masked reconstruction process.

G.4. Comparison of VAE decoders

Table 7 compares the effectiveness of two pre-training strategies for VAE decoders: MSE (Mean Squared Error) and EMA (Exponential Moving Average). The MSE approach results in an FID-15k score of 24.99, whereas the EMA method achieves a superior score of 24.66. This improvement underscores the benefits of using EMA, which enhances generative quality by fostering a more stable and consistent learning process for the VAE decoder. The slight yet significant reduction in FID suggests that EMA's smoothing effect on model parameters reduces variance in updates during training, mitigating the risk of convergence to suboptimal solutions. By averaging parameters over time, EMA offers a more reliable and steady representation, contributing to better convergence behavior and higher-fidelity image reconstructions.

G.5. Wider Masking ratio.

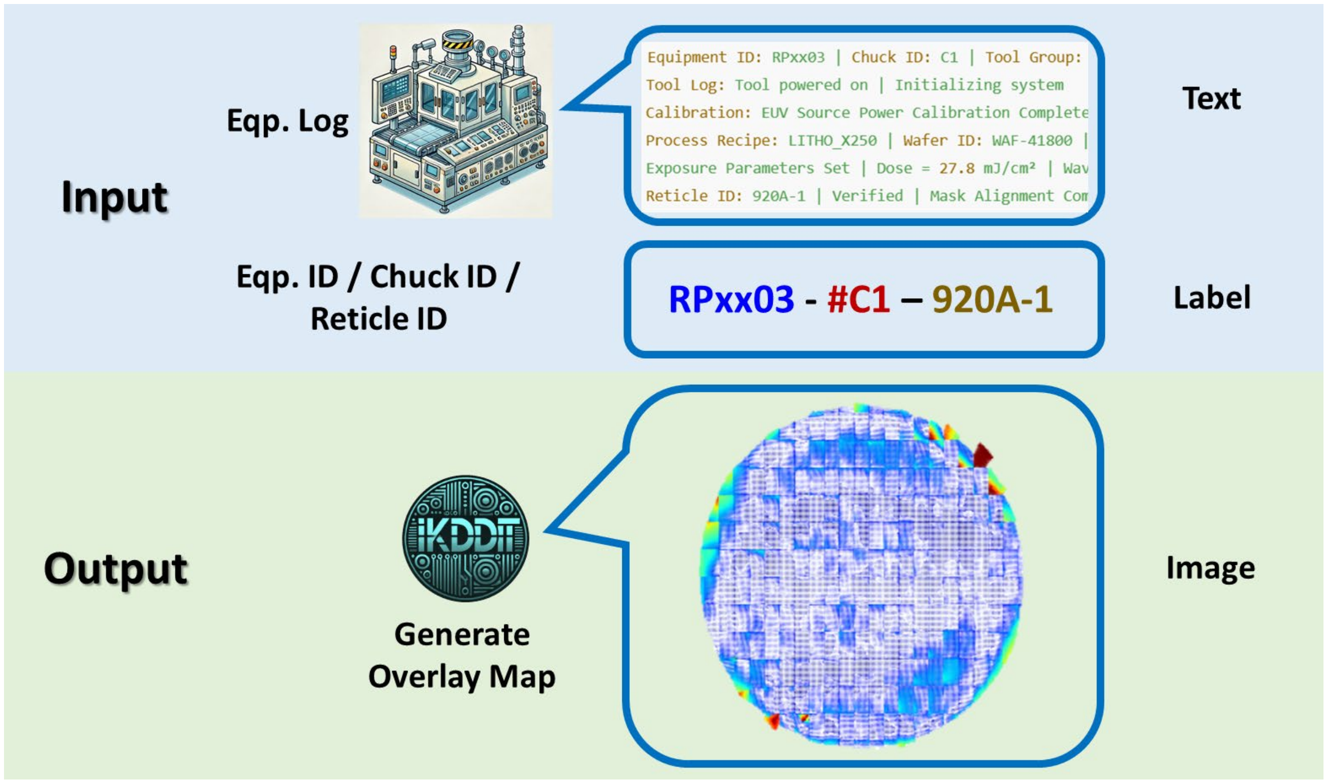
The masking ratio defines the proportion of input patches utilized during training. Table 8 and Figure 8 provide a comparative analysis of various masking ratios. For IKDDiT, a 50% masking ratio proves to be optimal, contrasting sharply with those commonly employed in recognition models, such as the 75% ratio used in MAE [9] and the 30% ratio in MDT [8]. We propose that image generation models necessitate more extensive information from a larger set of patches to produce high-quality outputs, while recognition models focus on the most crucial patches to extract semantic meaning effectively. This insight implies that using different masking ratios can enable the model to learn a more diverse range of contextual representations.

H. More Generated Samples

Figures 9 to 14 present representative samples of overlay map generation using our proposed IKDDiT model, conditioned on equipment logs. These equipment logs capture essential operational parameters and adjustments across various stages, enhancing the model's contextual understanding and enabling the synthesis of overlay maps that accurately reflect specific equipment configurations and conditions.

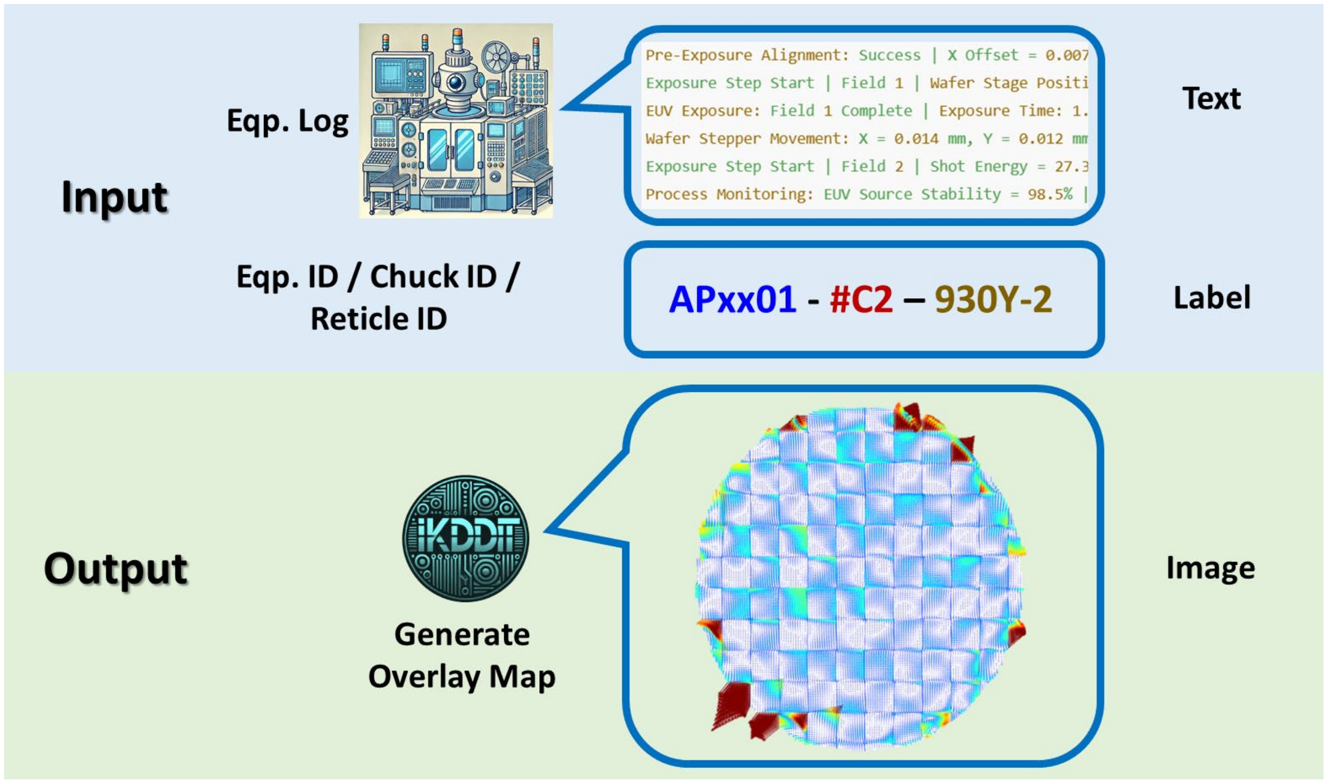
I. Detail Context of Equipment Logs

Figure 15 presents part of the equipment log during the photolithography process, detailing critical parameters such as exposure dose, focus offset, and stepper alignment values. These parameters play an essential role in maintaining precision across photolithography layers, as any deviations can directly impact overlay accuracy and, consequently, the yield. The log also records equipment IDs, wafer positions, and time stamps, which are essential for tracking equipment performance and identifying potential sources of overlay errors. By analyzing this log data, our model can gain insights into process variations and make more informed adjustments, thereby enhancing the overall accuracy of the overlay map predictions.



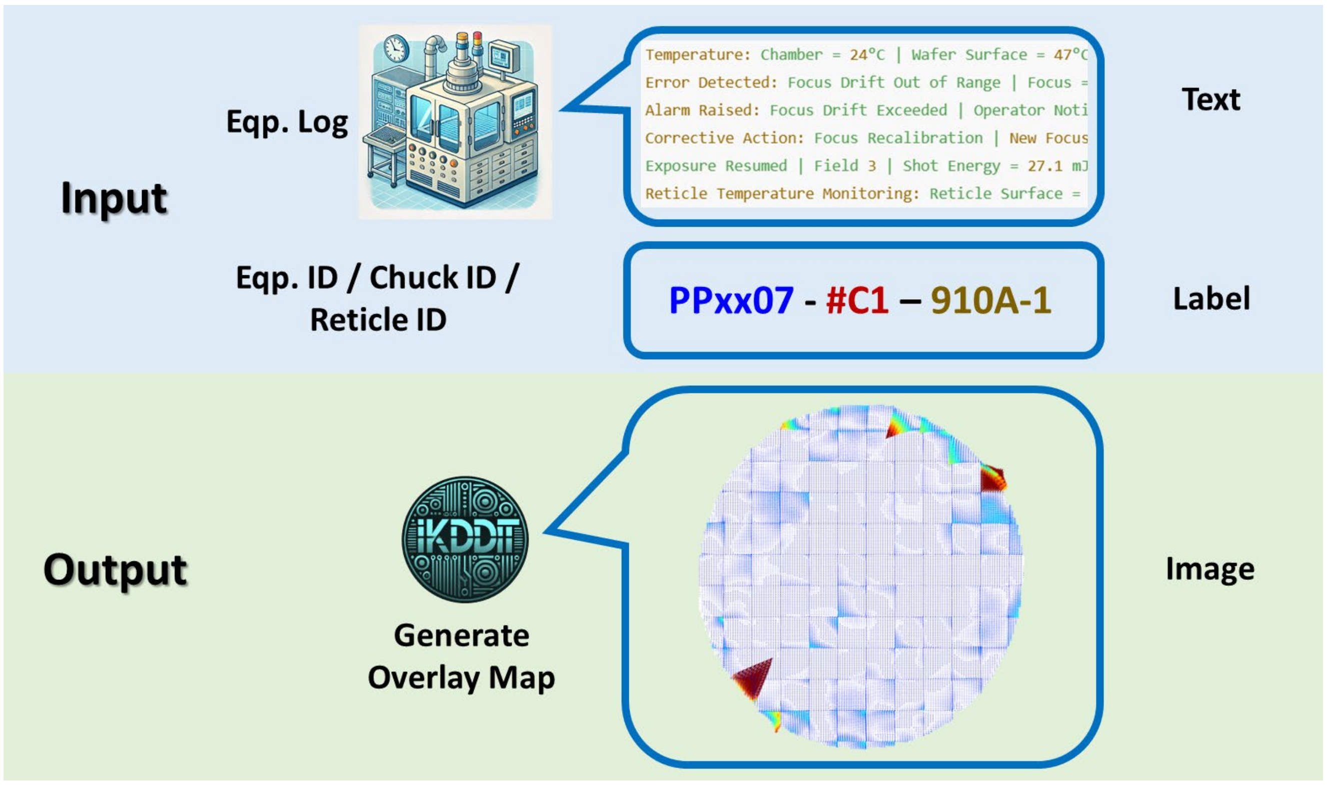
```
Equipment ID: RPxx03 | Chuck ID: C1 | Tool Group:
Tool Log: Tool powered on | Initializing system
Calibration: EUV Source Power Calibration Complete
Process Recipe: LITHO_X250 | Wafer ID: WAF-41800 |
Exposure Parameters Set | Dose = 27.8 mJ/cm² | Wav
Reticle ID: 920A-1 | Verified | Mask Alignment Com
```

Figure 9: Example of overlay map generated by IKDDiT under the condition of RPxx03-C1-920A-1, incorporating equipment logs.



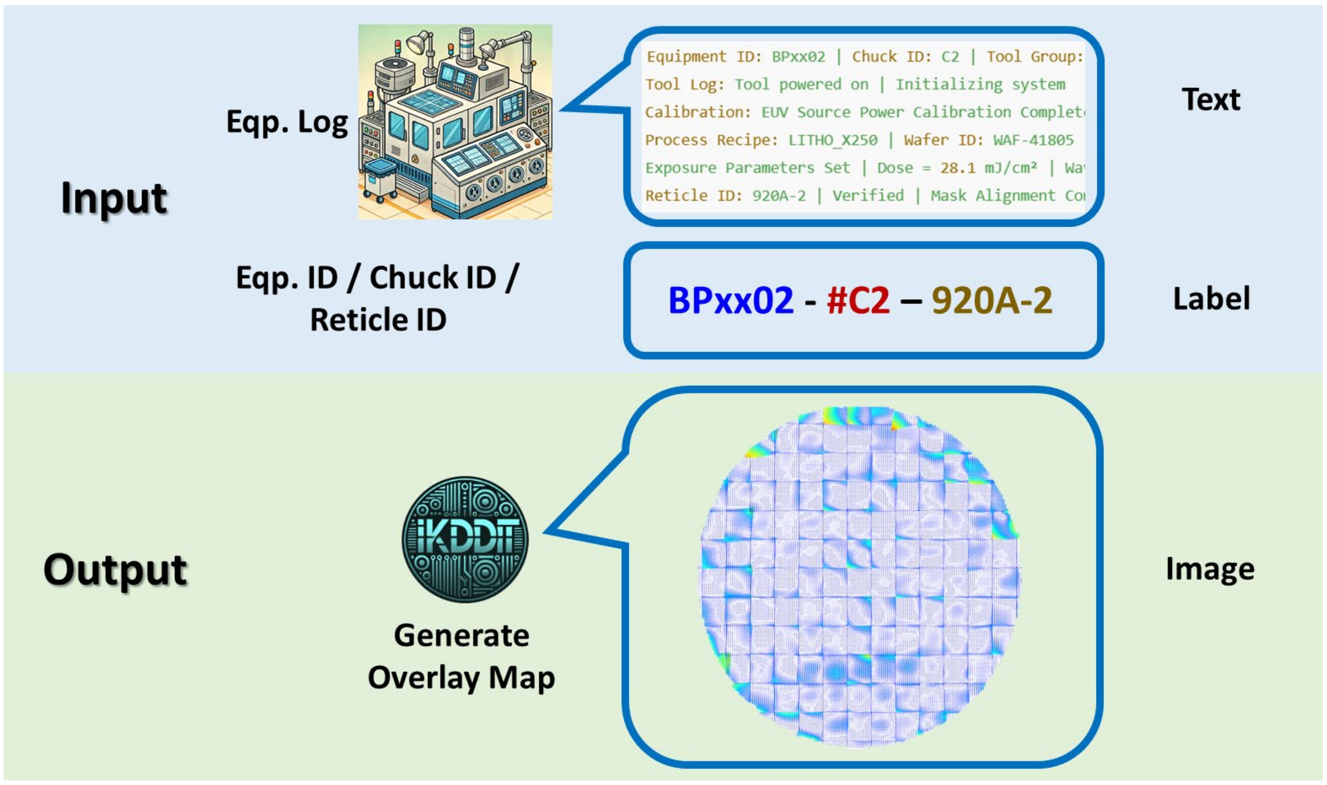
```
Pre-Exposure Alignment: Success | X Offset = 0.007
Exposure Step Start | Field 1 | Wafer Stage Positi
EUV Exposure: Field 1 Complete | Exposure Time: 1.
Wafer Stepper Movement: X = 0.014 mm, Y = 0.012 mm
Exposure Step Start | Field 2 | Shot Energy = 27.3
Process Monitoring: EUV Source Stability = 98.5% |
```

Figure 10: Example of overlay map generated by IKDDiT under the condition of APxx01-C2-930Y-2, incorporating equipment logs.



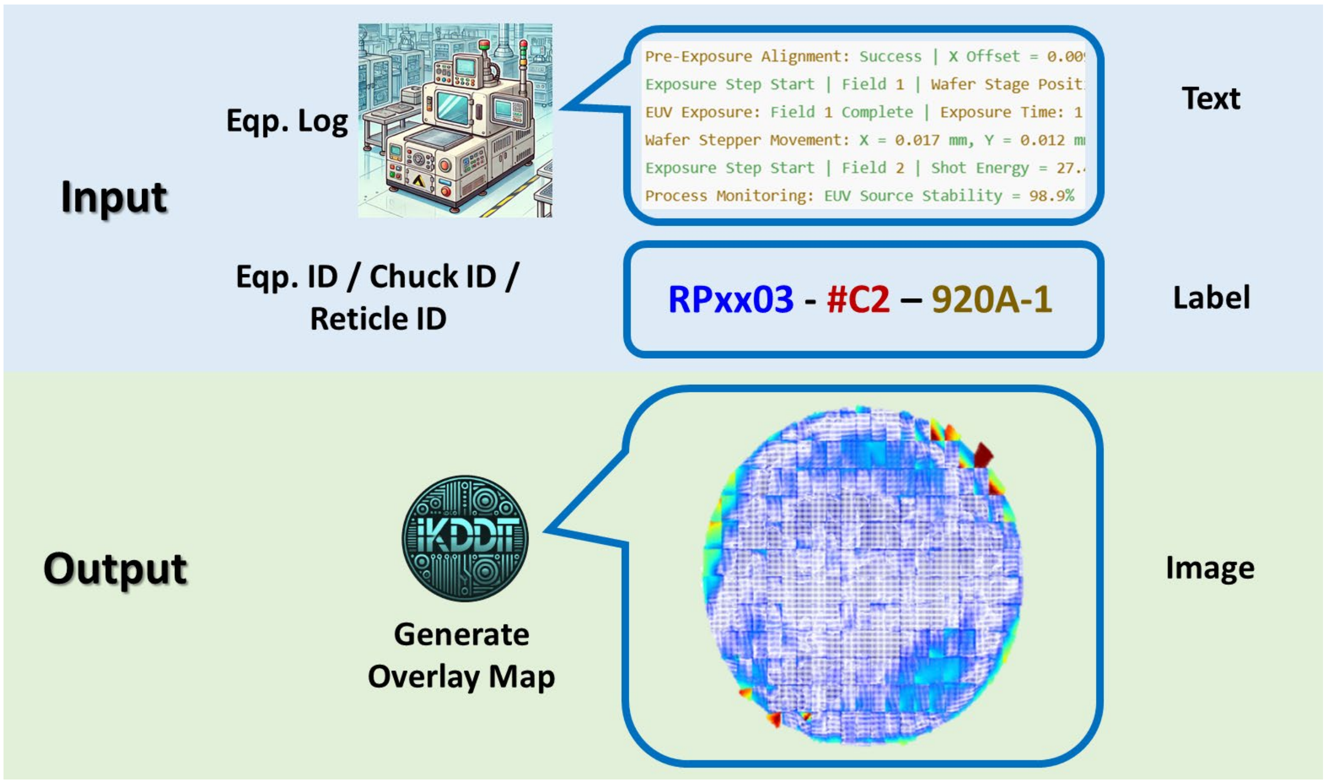
```
Temperature: Chamber = 24°C | Wafer Surface = 47°C
Error Detected: Focus Drift Out of Range | Focus =
Alarm Raised: Focus Drift Exceeded | Operator Noti
Corrective Action: Focus Recalibration | New Focus
Exposure Resumed | Field 3 | Shot Energy = 27.1 mJ
Reticle Temperature Monitoring: Reticle Surface =
```

Figure 11: Example of overlay map generated by IKDDiT under the condition of PPxx07-C1-910A-1, incorporating equipment logs.



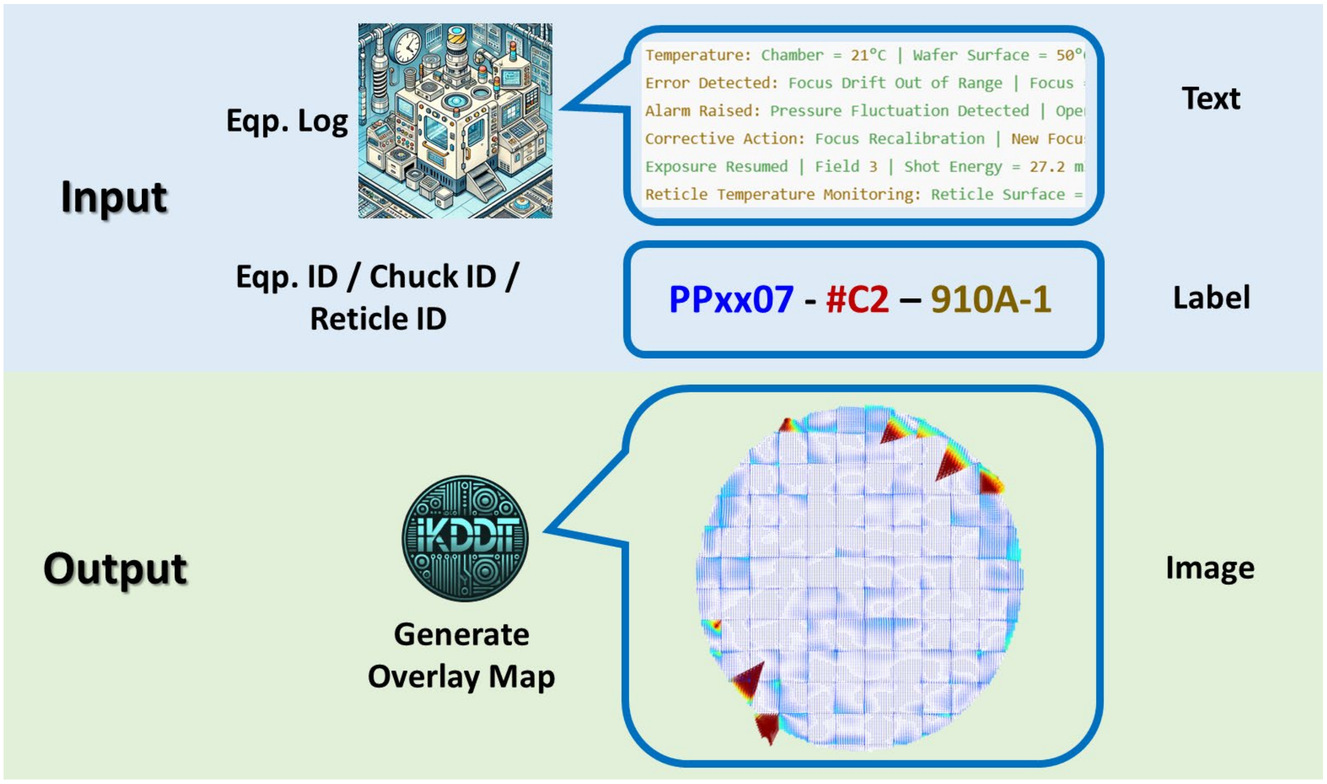
```
Equipment ID: BPxx02 | Chuck ID: C2 | Tool Group:
Tool Log: Tool powered on | Initializing system
Calibration: EUV Source Power Calibration Complet
Process Recipe: LITHO_X250 | Wafer ID: WAF-41805
Exposure Parameters Set | Dose = 28.1 mJ/cm² | Wa
Reticle ID: 920A-2 | Verified | Mask Alignment Co
```

Figure 12: Example of overlay map generated by IKDDiT under the condition of BPxx02-C2-920A-2, incorporating equipment logs.



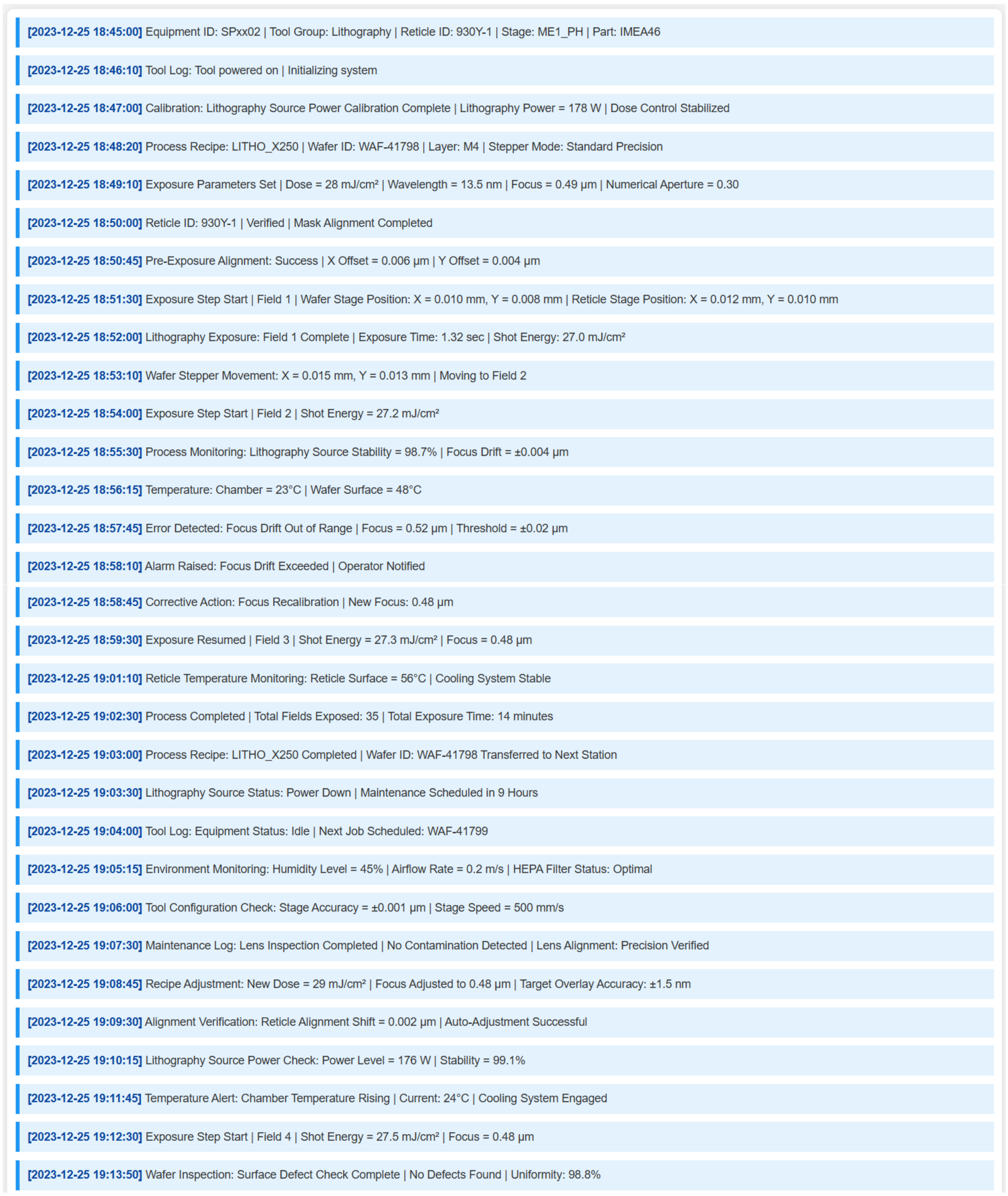
```
Pre-Exposure Alignment: Success | X Offset = 0.00
Exposure Step Start | Field 1 | Wafer Stage Posit
EUV Exposure: Field 1 Complete | Exposure Time: 1
Wafer Stepper Movement: X = 0.017 mm, Y = 0.012 m
Exposure Step Start | Field 2 | Shot Energy = 27.
Process Monitoring: EUV Source Stability = 98.9%
```

Figure 13: Example of overlay map generated by IKDDiT under the condition of APxx02-C1-920A-1, incorporating equipment logs.



```
Temperature: Chamber = 21°C | Wafer Surface = 50°
Error Detected: Focus Drift Out of Range | Focus
Alarm Raised: Pressure Fluctuation Detected | Ope
Corrective Action: Focus Recalibration | New Focu
Exposure Resumed | Field 3 | Shot Energy = 27.2 m
Reticle Temperature Monitoring: Reticle Surface =
```

Figure 14: Example of overlay map generated by IKDDiT under the condition of APxx03-C2-930Y-2, incorporating equipment logs.



```
[2023-12-25 18:45:00] Equipment ID: SPxx02 | Tool Group: Lithography | Reticle ID: 930Y-1 | Stage: ME1_PH | Part: IMEA46
[2023-12-25 18:46:10] Tool Log: Tool powered on | Initializing system
[2023-12-25 18:47:00] Calibration: Lithography Source Power Calibration Complete | Lithography Power = 178 W | Dose Control Stabilized
[2023-12-25 18:48:20] Process Recipe: LITHO_X250 | Wafer ID: WAF-41798 | Layer: M4 | Stepper Mode: Standard Precision
[2023-12-25 18:49:10] Exposure Parameters Set | Dose = 28 mJ/cm² | Wavelength = 13.5 nm | Focus = 0.49 µm | Numerical Aperture = 0.30
[2023-12-25 18:50:00] Reticle ID: 930Y-1 | Verified | Mask Alignment Completed
[2023-12-25 18:50:45] Pre-Exposure Alignment: Success | X Offset = 0.006 µm | Y Offset = 0.004 µm
[2023-12-25 18:51:30] Exposure Step Start | Field 1 | Wafer Stage Position: X = 0.010 mm, Y = 0.008 mm | Reticle Stage Position: X = 0.012 mm, Y = 0.010 mm
[2023-12-25 18:52:00] Lithography Exposure: Field 1 Complete | Exposure Time: 1.32 sec | Shot Energy: 27.0 mJ/cm²
[2023-12-25 18:53:10] Wafer Stepper Movement: X = 0.015 mm, Y = 0.013 mm | Moving to Field 2
[2023-12-25 18:54:00] Exposure Step Start | Field 2 | Shot Energy = 27.2 mJ/cm²
[2023-12-25 18:55:30] Process Monitoring: Lithography Source Stability = 98.7% | Focus Drift = ±0.004 µm
[2023-12-25 18:56:15] Temperature: Chamber = 23°C | Wafer Surface = 48°C
[2023-12-25 18:57:45] Error Detected: Focus Drift Out of Range | Focus = 0.52 µm | Threshold = ±0.02 µm
[2023-12-25 18:58:10] Alarm Raised: Focus Drift Exceeded | Operator Notified
[2023-12-25 18:58:45] Corrective Action: Focus Recalibration | New Focus: 0.48 µm
[2023-12-25 18:59:30] Exposure Resumed | Field 3 | Shot Energy = 27.3 mJ/cm² | Focus = 0.48 µm
[2023-12-25 19:01:10] Reticle Temperature Monitoring: Reticle Surface = 56°C | Cooling System Stable
[2023-12-25 19:02:30] Process Completed | Total Fields Exposed: 35 | Total Exposure Time: 14 minutes
[2023-12-25 19:03:00] Process Recipe: LITHO_X250 Completed | Wafer ID: WAF-41798 Transferred to Next Station
[2023-12-25 19:03:30] Lithography Source Status: Power Down | Maintenance Scheduled in 9 Hours
[2023-12-25 19:04:00] Tool Log: Equipment Status: Idle | Next Job Scheduled: WAF-41799
[2023-12-25 19:05:15] Environment Monitoring: Humidity Level = 45% | Airflow Rate = 0.2 m/s | HEPA Filter Status: Optimal
[2023-12-25 19:06:00] Tool Configuration Check: Stage Accuracy = ±0.001 µm | Stage Speed = 500 mm/s
[2023-12-25 19:07:30] Maintenance Log: Lens Inspection Completed | No Contamination Detected | Lens Alignment: Precision Verified
[2023-12-25 19:08:45] Recipe Adjustment: New Dose = 29 mJ/cm² | Focus Adjusted to 0.48 µm | Target Overlay Accuracy: ±1.5 nm
[2023-12-25 19:09:30] Alignment Verification: Reticle Alignment Shift = 0.002 µm | Auto-Adjustment Successful
[2023-12-25 19:10:15] Lithography Source Power Check: Power Level = 176 W | Stability = 99.1%
[2023-12-25 19:11:45] Temperature Alert: Chamber Temperature Rising | Current: 24°C | Cooling System Engaged
[2023-12-25 19:12:30] Exposure Step Start | Field 4 | Shot Energy = 27.5 mJ/cm² | Focus = 0.48 µm
[2023-12-25 19:13:50] Wafer Inspection: Surface Defect Check Complete | No Defects Found | Uniformity: 98.8%
```

Figure 15: The examples of equipment log in photolithography process.

References

[1] Hsiu-Hui Hsiao and Kung-Jeng Wang, "GAGAN: global attention generative adversarial networks for semiconductor advanced process control", *IEEE Transactions on Semiconductor Manufacturing*, vol. 37, no. 1, pp. 115-123, Feb. 2024. 1.

[2] William Peebles and Saining Xie, "Scalable diffusion models with transformers", *IEEE/CVF International Conference on Computer Vision (ICCV)*, Paris, France, pp. 4172-4182, 2023. 2, 5.

[3] Prafulla Dhariwal and Alexander Nichol, "Diffusion models beat gans on image synthesis", *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. 2.

[4] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine, "Elucidating the design space of diffusion-based generative model", *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. 2.

[5] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin, "Attention is all you need", *Advances in Neural Information Processing Systems (NeurIPS)*, 2017. 2, 5.

[6] Hongkai Zheng, Weili Nie, Arash Vahdat, and Anima Anandkumar, "Fast training of diffusion models with masked transformers", *arXiv preprint arXiv:2306.09305*, 2023. 2.

[7] Rui Zhu, Yingwei Pan, Yehao Li, Ting Yao, Zhenglong Sun, Tao Mei, and Chang Wen Chen, "SD-DiT: Unleashing the power of self-supervised discrimination in diffusion transformer", *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. 2.

[8] Shanghua Gao, Pan Zhou, Ming-Ming Cheng, and Shuicheng Yan, "Masked diffusion transformer is a strong image synthesizer", *arXiv preprint arXiv:2303.14389*, 2023. 2, 9.

[9] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick, "Masked autoencoders are scalable vision learners", *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 16000–16009, 2022. 2, 9.

[10] Jinghao Zhou, Chen Wei, Huiyu Wang, Wei Shen, Cihang Xie, Alan Yuille, and Tao Kong, "ibot: Image bert pre-training with online tokenizer", *International Conference on Learning Representations (ICLR)*, 2022. 3.

[11] Mathilde Caron, Hugo Touvron, Ishan Misra, Herv´e J´egou, Julien Mairal, Piotr Bojanowski, and Armand Joulin, " Emerging properties in self-supervised vision transformers", *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021. 3.

[12] Ilya Loshchilov and Frank Hutter, "Decoupled weight decay regularization", *International Conference on Learning Representations (ICLR)*, 2019. 3.

[13] Ilya Loshchilov and Frank Hutter, "SGDR: Stochastic gradient descent with warm restarts", *International Conference on Learning Representations (ICLR)*, 2017. 3.

[14] Jonathan Ho, Ajay Jain, Pieter Abbeel, "Denoising Diffusion Probabilistic Models", *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 3, 4.

[15] Alexander Quinn Nichol and Prafulla Dhariwal, "Improved denoising diffusion probabilistic models", *International Conference on Machine Learning (ICML)*, 2021. 4.

[16] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Bjorn Ommer, "High-resolution image synthesis with latent diffusion models", *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10684–10695, 2022. 4, 5.

[17] Jonathan Ho and Tim Salimans, "Classifier-free diffusion guidance", *Advances in Neural Information Processing Systems (NeurIPS), Workshop on Deep Generative Models and Downstream Applications*, 2021. 4.

[18] Huiwen Chang, Han Zhang, Jarred Barber, AJ Maschinot, Jose Lezama, Lu Jiang, Ming-Hsuan Yang, Kevin Murphy, William T Freeman, Michael Rubinstein, et al., "Muse: Text-to-image generation via masked generative transformers", *arXiv preprint arXiv:2301.00704*, 2023. 4, 8.

[19] Jianwei Yang, Chunyuan Li, Pengchuan Zhang, Bin Xiao, Ce Liu, Lu Yuan, and Jianfeng Gao, "Unified contrastive learning in image-text-label Space", *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 6.

[20] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei, "Scaling laws for neural language models", *arXiv preprint arXiv:2001.08361*, 2020. 7.